Preprint
UCRL-JC-149655

Illumination Under Trees

N. Max

This article was submitted to the Human and Computer 2002, Aizu, Japan, September 11-14, 2002

August 19, 2002

U.S. Department of Energy

Lawrence
Livermore
National
Laboratory

Illumination under Trees

Nelson Max
Lawrence Livermore National Laboratory
e-mail: max2@llnl.gov

Abstract

This paper is a survey of the author's work on illumination and shadows under trees, including the effects of sky illumination, sun penumbras, scattering in a misty atmosphere below the trees, and multiple scattering and transmission between leaves. It also describes a hierarchical image-based rendering method for trees.

Keywords: penumbras, multiple scattering, radiosity, atmospheric illumination, shadows, hierarchical models, image-based rendering.

1. Introduction

Illumination effects under a canopy of trees are very complex, due to shadowing and interreflection from many leaves and branches. This paper is a survey of my work in this area, including the effects of sky illumination, sun penumbras, scattering in a misty atmosphere below the trees, and multiple scattering and transmission between the leaves.

The input to these calculations is a geometrical model for the trees. Impressively detailed and beautiful models have been produced by Prusinkiewicz *et al.* [1 - 5], de Reffye *et al.* [6], Weber and Penn [7], and Reeves and Blau[8]. In this work, I use models from Bloomenthal [9], Lintermann and Deussen[10], and Ohsaki *et al.* [11, 12], as well as a hierarchical model of my own, described in section 2 below.

For the basic shadowing, I use variants of the shadow volume algorithm of Crow [13], the Z buffer algorithm of Williams [14], and the ray tracing algorithm of Whitted [15]. Section 3 describes the atmospheric illumination in the mist below the trees, sections 4 and 5 describe two methods of generating the penumbras from the sun's disc, and section 6 describes an approximation to the multiple scattering among the leaves.

2. Hierarchical Image-Based Modelling

Since trees are complicated to model and render with polygons, an alternative is image-based modelling, in which color/depth images from a small collection of input views are reprojected for a new viewpoint. Chen and Williams [16] first did this reprojection based on image flow over large smooth or planar regions.

However, for trees with many small leaves, there are no large regions with coherent depth, so I chose to reproject each pixel separately, as described in [17]. I precomputed the input images in orthogonal views, as RGBα images without lighting, but with a depth and normal vector at each pixel. Each non-transparent pixel corresponds to a 3D point, whose x and y coordinates are determined by the pixel location and whose z coordinate is determined from the depth. This point is reprojected in perspective into a new view, using a 4 by 4 transformation matrix which is the product of the viewing matrix for the new view and the inverse of the viewing matrix for the input view. The normal is also rotated into the new view with the upper left 3 by 3 rotation part of the transformation matrix, for shading in a new lighting situation.

One innovation was to store information for surfaces at multiple depths along the viewing ray through each pixel, as in the A buffer of Carpenter [18]. This allows surfaces which might otherwise not have been visible in an input view to be reprojected into a new view where they become visible, solving the "disocclusion" problem. A similar idea was later used in the Layered Depth Image (LDI) of Shade *et al.* [19]. I also reprojected several of the nearest input views, to better cover the new view and avoid missed "hole" pixels.

Figure 1 shows such a reprojected view of an *Abies sachalinensis* Masters tree as modelled in [11],

and figure 2 shows several *Magnolia obavata* Thunberg trees, as modelled in [12]. Shadows were added to figure 2 with the Z buffer shadow algorithm of Williams [14], by reprojecting the same 3D points into an orthogonal image from the point of view of the sun, and using the depths in this image for comparison to see which surface points are visible from the sun.

Figure 1. A reprojected *Abies* tree. Reprinted with permission from [17].

Figure 2. A grove of seven identical rotated and translated *Magnolia* trees, with shadows. Reprinted with permission from [17].

These reprojection methods [17, 19] are good for viewing a tree from outside, but if the perspective viewpoint moves too close, missed pixels show up between the reprojected pixels. These holes can be filled by reprojecting each input pixel into a larger block of output pixels, but this results in a blocky appearance to the output. Therefore in [20] I developed a hierarchical reprojection algorithm, where the tree geometry is based on a hierarchy of objects of different sizes, each with its own set of precomputed views. The highest level of detail is the single scanned maple leaf shown in figure 3, as an RGBα texture in a single rectangle. Figure 4 shows a branch at an intermediate level of detail, and figure 5 shows the whole tree. Figure 6 shows a view from near the tree, using reprojections of objects at different levels of detail and resolutions, as required to get the proper resolution of each object for the new viewpoint. Figure 7 shows a grove of several of these maple trees, with the leaves recolored in different fall colors.

Figure 3. A maple leaf. Reprinted with permission from [20].

Figure 4. A small branch. Reprinted with permission from [20].

Figure 5. A reprojected maple tree. Reprinted with permission from [20].

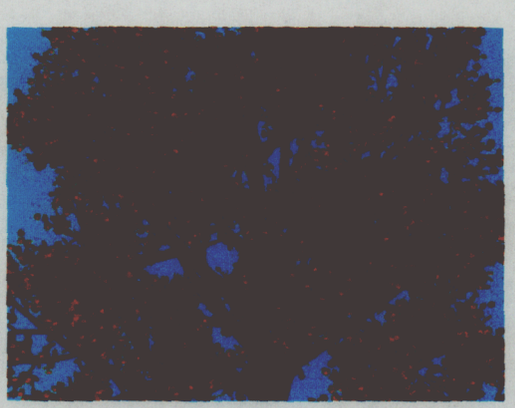

Figure 6. The maple tree reprojected from a closer viewpoint. Reprinted with permission from [20].

Figure 7. A grove of 7 reprojected and recolored maple trees. Reprinted with permission from [20].

Pseudocode for recursive choice of the appropriate level objects to reproject, and the computation of the corresponding reprojection matrices, is shown below. The constant `t` is a threshold above which reprojected pixels will be too far apart and cause holes, so that a higher level of detail is required. The polygons in the object structures define the branches at the current hierarchical level, to which the smaller branch or leaf objects are attached, using transformation matrices to move them from their standard model coordinates into position on the branch. The matrix multiplication `Q*S.matrix` recursively multiples these transformation matrices together to get the final model-to-view matrix for the object.

```
struct object {
  bounding_sphere;
  list_of_polygons;
  list_of_subobjects, with matrices;
  }
Decide (object K, matrix Q) {
  if(K.bounding_sphere ovelaps view volume)
    if(projected size of closest pixel < t)
      Reproject (K, Q);
    else {
      for polygons P in K.list_of_polygons
        render P using viewing matrix Q;
      for objects S in K.list_of_subobjects
        Decide (S, Q*S.matrix);
      }
  }
```

In [21] I applied the alpha-test techniques of Schauffler [22] to do the hierarchical reprojection in hardware. The depth is transferred into the α component of an RGBα texture, which is used on a series of rectangles through the object volume, parallel to the image plane of the input view. The alpha test of OpenGL is used to select only those fragments of the textured polygon whose α-coded depths lie within an appropriate depth slab of each of these rectangles. When the textured rectangles are transformed into their correct positions in the new view, this approximates the per-pixel reprojection. This method cannot reproject multiple surfaces at the same pixel or texel, so I sliced the tree volume in each input view into several large slabs, each with its own texture.

I also used the OpenGL color matrix in a second pass to rotate the normals (encoded in a second RGBα image) into the new view, to take the dot product with the light vector, and to create a black and white shading image, as suggested by Westermann and Ertl [23]. The alpha test was again applied to gen-

erate the same fragments as in the unshaded color image. Finally I multiplied the unshaded color image by the shading image to get a shaded view, using OpenGL image copy and blending functions.

I revised the recursive traversal algorithm in the above pseudocode to collect the model/view matrices for all needed instances of each input image, and then do all reprojections of the same image in sequence, so that each texture is loaded only once per frame. Nevertheless, I only got a speed up factor of 5 over the software method, which was not enough for real time motion through forests with multiple trees. Figure 8 shows an image created with this hardware method, using both the hierarchical maple tree described above and a hierarchical oak tree created by Oliver Deussen using the method of [10].

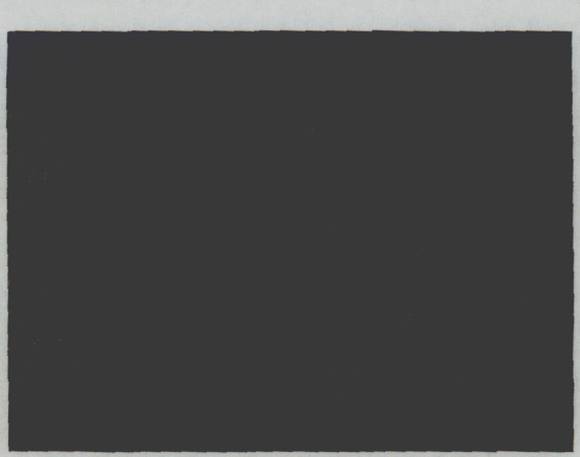

Figure 8. A mixed forest of oak and maple trees, reprojected in hardware.

3. Atmospheric Illumination and Shadows

In order to compute the effects of scattering from mist droplets, the illuminated and shadowed segments along each viewing ray must be found, and then the scattered illumination can be computed by integration along the illuminated segments. To do this, I used the shadow volume algorithm of Crow [13], which augments the model geometry with extra transparent shadow polygons. These are semi-infinite wedges, bounded by a profile edge of the input model, and the extension of the rays between the light source and the edge endpoints, as shown in figure 9.

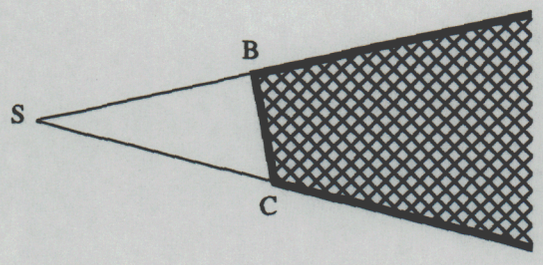

Figure 9. The semi-infinite shadow polygon cast by profile edge BC and the light source S.

If a viewing ray to a point *P* crosses a front-facing shadow polygon, it enters a shadow, and a back-facing shadow polygon can cancel this effect. So a count is initialized with the number of scene polygons shadowing the viewpoint, incremented for each front facing polygon crossed, and decremented for each back-facing one. If the result is positive, *P* is in shadow. This shadow algorithm has now been implemented in hardware, using the OpenGL stencil buffer to count the shadow polygons crossed. If instead, all the intersections of the viewing ray with the shadow polygons are explicitly computed in software, they can be used to determine the segments that are illuminated, and thus the atmospheric scattering. Figure 10 was created in this way, and shows the illuminated and shadowed volumes as beams diverging from the projection of the sun.

Figure 10. Atmospheric illumination under three maple trees.

The leaves were modeled as 14 sided polygons, instead of as the RGBα textures described in section 2 above, in order to generate the correct shadow polygons. This defines a very large number of shadow polygons, so instead of generating them and including them in my software scan line algorithm, I created them on the fly as needed.

If I had used the traditional horizontal scan lines, then the scan planes through the eye and the scan lines in the lower half of the image would intersect almost all the shadow polygons, slowing down the algorithm. Instead, I used polar coordinates, with radial scan lines radiating from the projection of the light source. These are the projections on the image plane of scan planes through the viewpoint and the light source, as shown in figure 11. The advantage is that the only shadow polygons that can affect such a scan plane are cast by polygon edges which intersect that scan plane.

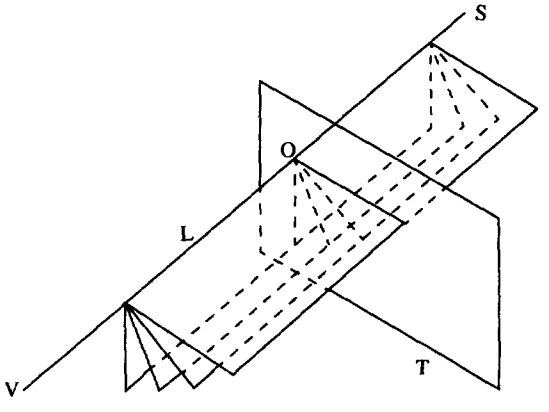

Figure 11. Four scan planes, meeting at a line L between the viewpoint V and the light source L. They intersect the view plane T in four radial scan lines from the point O where L intersects T.

I generated the shadow rays in which these shadow polygons intersect the scan plane by processing the opaque polygon edges intersecting the scan plane in radially increasing order along the scan line, in a variant of the Watkins span-coherence hidden surface algorithm [25]. As shown in figure 12, this means that at the time a visible span segment of an opaque polygon is shaded, the shadow rays in front of it have already been created, and can be used to determine the illuminated segments along the viewing rays, as well as whether the span is in shadow. (If the light source is on the opposite side of the viewpoint as the view plane, the processing must be done in radially decreasing order instead.)

Figure 10 was resampled from polar coordinate scan segments into a normal raster image using the algorithm of [26]. It is a frame from the animation [27]. The shadows on the bumps in the tree bark were determined by horizon mapping [28].

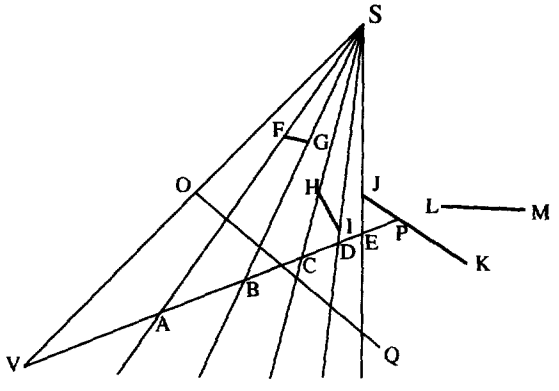

Figure 12. A scan plane containing the viewpoint V and the light source S, intersecting the opaque input polygons in segments FG, HI, JK, and LM, and the view plane in radial scan line OQ. The polygon edge intersections with this scan plane, in outwards radial order of their projections on OQ, are F, G, H, I, J, L, M, and K. At the time surface point P on segment JK is shaded, the shadow rays SFA, SGB, SHC, SID, and SJE have been created as the radial scan passed points F, G, H, I, and J, and these shadow rays mark off the illuminated segments VA, BC, and DE on the viewing ray VP. Since there are three front-facing and two back-facing shadow ray intersections, and the viewpoint V is not itself in shadow, P is in shadow.

4. Sky Illumination and Sun Penumbras

The shadows on the ground in figure 10 have sharp edges separating light and shadow regions of constant color. This is incorrect for two reasons; the finite sized disc of the sun should create a shadow penumbra, and the effects of illumination and shadowing of the sky hemisphere should cause shading variation even in the umbra and fully lit regions.

Shadow penumbras and sky illumination were introduced to computer graphics by Nishita and Nakamae in [29] and [30] respectively. In [31], I combined these effects in a uniform way using the concept of the convolution and the Fast Fourier Transform (FFT).

I assumed that the sky (with clouds) and the sun were on a sphere at infinity, approximated by a finite sized rectangle T of direction vectors in the plane $z = 1$, rather than the whole infinite plane, and that the leaf canopy can be approximated by a shadow mask in a single plane at the average vertical height h of the leaves. In [31], I analyzed the errors introduced by these two assumptions, and concluded that they were small.

The shadow mask $M(x, y)$ in the plane $z = h$ is just a translated version of the shadows on the ground shown in figure 10. The mask is zero where a parallel source sun with an infinitessimal disc would be blocked, and the mask is one where the light would get through. The sun plus sky radiance is a function $R(\omega)$, where $\omega = (u, v, w)$ is a unit direction vector. We can also think of the radiance as a function $J(x, y)$ defined on rectangle S in the plane $z = h$, which is a scaled version of T, that is, $R(\omega) = J(x, y)$ with $(x, y) = (hu/w, hv/w)$.

For a specific origin point O on the ground, the non-occluded part of the sun/sky radiance is then the product $M(x, y)\, J(x, y)$. For any other point $P = O + (k, l)$, the tree canopy mask appears translated but the illumination from the sun and sky does not, so P sees radiance $R(\omega) = M(x + k, y + l)\, J(x, y)$. The irradiance $I(P)$ at P, used to determine the diffuse shading, is

$$I(P) = \int_{\Omega} R(\omega)\cos(\theta)\, d\omega$$

where Ω is the unit hemisphere above the ground, and θ is the angle between ω and the vertical surface normal. By a change of variables (see [31] or [32]), and by approximating the integral over the infinite plane by the integral over the rectangle S, we have

$$I(O + (k, l)) \approx \iint_{S} \frac{M(x + k, y + l)J(x, y)}{(x^2 + y^2 + h^2)^2}\, dx\, dy$$

which is a correlation between $M(x, y)$ and $K(x, y) = J(x, y) / (x^2 + y^2 + h^2)^2$, or else a convolution of $M(x, y)$ and $K(-x, -y)$. If M and K are both represented as n by n rasters, this convolution integral can be computed with n^2 multiplications and n^2-1 additions, so the time to compute $I(P)$ for all n^2 points $P = O + (k, l)$ in an n by n raster would be $O(n^4)$.

By the convolution theorem, this convolution can instead be computed for all n^2 points P by taking the Fourier transforms of M and K, multiplying them, and then taking the inverse Fourier transform. If each of the necessary 2D Fourier transforms are done by applying the $O(n \log n)$ cost 1D FFT algorithm $2n$ times, first on the n rows and then on the n columns, the total cost is $O(n^2 \log n)$, which is much faster than $O(n^4)$.

Figure 13 shows a raster image of the shadows on the ground plane, with penumbras and sky illumination, and figure 14 shows the final result, with these shadows texture-mapped onto the ground plane. Note the blurred shadow penumbra from the sun, and the fact that the shading is not constant even in the umbra and fully lit regions. (The sun at a slightly different position than in figure 10.) In the animation "Sun and Shade" [33], the clouds move in front of the sun, and the sun illumination decreases non-monotonically until only the sky illumination is left. The light beam effect in figure 10 was added, proportional to the sun illumination.

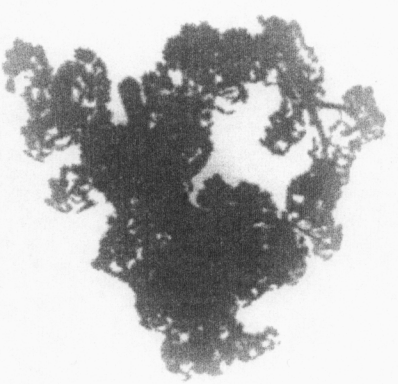

Figure 13. Shadow image with sky illumination and sun penumbra effects. Reprinted with permission from [31].

As explained above, the FFT method is only correct for shadows cast by a mask object in one plane onto a parallel receiving plane, so the width of the shadow penumbra from the tree trunk onto the ground is constant, which is not correct. In [31] I suggested how this problem might be corrected using multiple shadow casting planes, but did no implementation. Soler and Sillion [34, 35] describe a complete system for choosing multiple mask and receiving planes to create shadows which are correct to within a given error tolerance.

Figure 14. Shadows of figure 13, drawn on the ground plane. Reprinted with permission from [31].

5. Penumbras from Layered Depth Images

Another method to give accurate penumbra width variation for objects like tree trunks is to use distributed ray tracing, as in [36], tracing rays from various points on the surface area projecting onto the pixel to various points on (or directions to) the light source. For a complex tree or forest model, this could be very expensive. On the other hand, if we use layered depth images as in section 2, the data size is more independent of the model complexity. Lischinski and Rappoport [37] were the first to do ray tracing in an LDI, for the purposes of reflections in shiny or glossy surfaces. In [38], Brett Keating and I optimized ray tracing in an LDI as applied to shadow penumbras.

The scene geometry is converted into an LDI from the point (or direction) of view of the light source. The model is sliced by equally spaced planes perpendicular to the lighting direction, and the LDI is further discretized into a 32 bit word at each pixel, with each bit representing the presence of a surface in a corresponding slab between two slice planes. Shadow rays are traced through this discretized LDI. To eliminate light leaks between the LDI fragments of steeply sloped polygons, the shadow rays are actually intersected with each slice plane, and considered obscured if the bits at the intersected pixel are set for either of the slabs separated by the slice plane. The (possibly weighted) fraction of non-obscured rays determines the penumbra illumination. Details of how this is done efficiently using logical operations on the 32 bit words are given in [38].

This LDI ray tracing is supplemented by the Z-buffer shadow algorithm [14] and its "percentage close filtering" variant [39], applied to the non-discretized LDI surfaces in the two slabs nearest to the surface point being shaded along the direction to the light source, in order to get good anti-aliased shadows. Figure 15 shows this method applied to a leafless bush. Note the smoothly varying width of the penumbras.

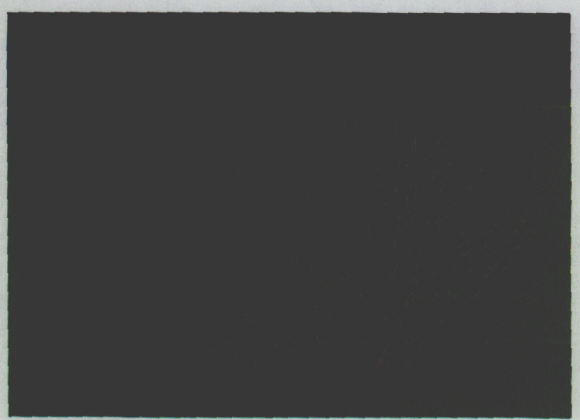

Figure 15. Shadow penumbras cast from a leafless bush.

Instead of tracing shadow rays from random points on the surface area projecting to the pixel to random points on the light source, effectively doing 4D Monte Carlo integration, we traced rays from random points on the surface area to deterministic points on the light source corresponding to scaled versions of the surface positions, thus doing 2D Monte Carlo in-

tegration. The result is a biased estimate of the correct 4D integral, but with the same number of rays, the 2D integral has less variance, so the images appear less noisy, and look better. Figure 16 shows a tree with penumbras on the shadows cast from the upper leaves onto lower ones, as well as onto the ground, as computed with this method.

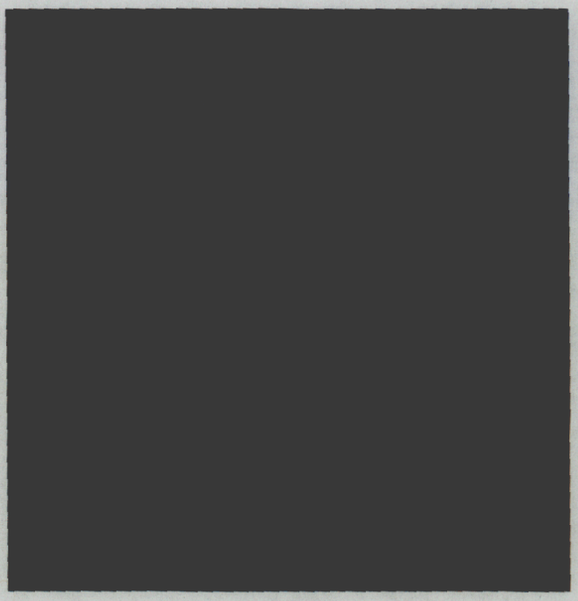

Figure 16. Shadows from a tree, with penumbras. Reprinted with permission from [38].

6. Plane Parallel Radiance Transport

Below a dense tree canopy, very little of the light is direct illumination; most of the light arriving near the ground has been scattered multiple times by the leaves. The radiosity method [32, 40] accounts for multiple diffuse scattering by solving a system of m linear energy balance equations in the m unknown radiosities, one for each surface. In a forest environment with many hundred thousand leaves, computing and even storing the matrix for this system would be impractical. Therefore, I instead approximated the scattering from the leaves, twigs, and branches by a distribution of infinitessimal scattering particles, like a cloud of smoke. For simplicity, I made the "plane parallel" assumption that the density and scattering properties of these particles vary only with the height above the ground, and do not vary with the two horizontal coordinates.

The particles were not isotropic scatterers. Instead their scattering phase functions of incident and reflection angles were computed from a distribution of surface normals and from the bidirectional reflection distribution function (BRDF) and the bidirectional transmission distribution function (BTDF) of the surfaces. To build the distribution of normals, the geometry of the forest model is sliced into horizontal sections, and in each section, the surface area for surface normals in each of a collection of direction bins is computed. This is done by clipping the geometry both by height and surface normal. I made a further symmetry assumption that the distribution of normals is independent of the azimuthal angle (orientation of the horizontal component) of the surface normal, so the normal bins are indexed only by the angle between the surface normal and the vertical.

The general equation of radiance transport is a differential-integral equation which expresses the change in radiance flowing along a ray as the sum of an extinction term representing blockage by the particles, and a source term representing the inscattering. The source term is an integral, over the direction sphere, of the radiance in each incoming direction times the scattering phase function giving the probability that light in this incoming direction will scatter into the ray in question. If the light flow directions are discretized into direction bins, the extinction and source terms are represented by matrices which vary with position, and we get a coupled system of linear partial differential equations for the radiance of each direction bin as a function of position. The plane parallel assumption reduces this to an ordinary differential equation (ODE) system in the single height variable z. The unknowns are a vector $I(z)$ of radiances, one for each direction bin.

It is useful to split $I(z)$ into two vectors, $I_u(z)$ for the direction bins for upward flowing radiance, and $I_d(z)$ for the downwards ones. We have "two point" boundary conditions for the ODE system: (1) $I_d(h)$ equals the sun and sky illumination at the height h of top of the canopy, and (2) the BRDF of the ground surface determines $I_u(0)$, as a function of $I_d(0)$. When

the radiance is discretized into direction bins, the BRDF of the ground is represented by a matrix F, with $I_u(0) = F\, I_d(0)$.

The standard ODE methods like Runga-Kutta only apply to initial value problems where all components of the vector of unknowns are determined at one boundary point. In order to generate an initial value problem, we write new equations for a matrix $F(z)$ representing the BRDF of at height z, including the effects of multiple reflections within the slab of the leaf canopy between heights 0 and z. The BRDF of the ground determines the initial condition $F(0)$, so we can find $F(z)$ for increasing z by the Runga-Kutta method. The matrix $F(h)$ determines $I_u(h)$ from $I_d(h)$. Then we know the initial values of both $I_d(z)$ and $I_u(z)$ at $z = h$, and can solve for them for decreasing z by the Runga-Kutta method. The mathematical details and equations are given in [41] and [42]. The plane-parallel radiance calculations were carried out using a modified version of the Hydrolight radiative transfer code [www.sequoiasci.com].

Figure 17. View at top of tree canopy, showing diffuse sphere. Reprinted with permission from [41].

The trees and bamboo in figures 17, 18, and 19 were made using the hierarchical modelling techniques of section 2. One of the objects was a collection of trees or bamboo which was repeated in a periodic tiling.

A small diffuse sphere, of 79% reflectivity at all wavelengths, was inserted to show the local illumination near the viewpoint. In Figure 17, the bottom of the sphere shows the effects of the upward flowing radiance from the leaf canopy.

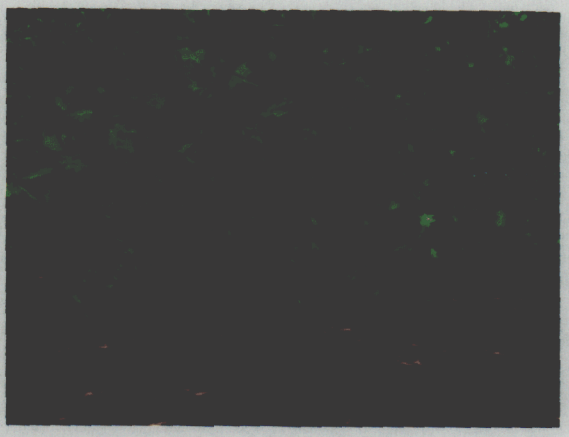

Figure 18. View of trees and diffuse sphere near the ground. Reprinted with permission from [41].

Figure 18 is a view near the ground. The illumination is much less than in figure 17, so an automatic exposure compensation was used. This is why the sky and ground colors look different in theses two images. The texture apparent on the ground show shadows from the sun, created by a percentage closer Z buffer shadow algorithm [14, 39]. In order not to double-count the direct solar illumination, its contribution to the ODE solution was identified and subtracted. The shading on the surfaces was done by multiplying the (shadowed) direct solar illumination plus the adjusted ODE solution by the surface BRDF/BTDF and the cosine of the angle of incidence, and integrating the product over the unit sphere (*i. e.* summing over the direction bins). A total of 434 direction bins were used, so this sum was precomputed for various surface orientations at sampled heights above the ground, and then interpolated for faster shading. See [41] for details.

Figure 19 shows a bamboo grove on an overcast day, without shadows.

Figure 19. A bamboo grove on an overcast day, without shadows. Reprinted with permission from [41].

Acknowledgments

This work was performed under the auspices of the U. S. Department of Energy by the University of California, Lawrence Livermore National Laboratory under contract No. W-7405-Eng-48. I wish to thank the coauthors of the papers I have surveyed here: Keiichi Ohsaki, Brett Keating, Curtis Mobley, Oliver Deussen, and Eh-hua Wu, for their input, Jules Bloomenthal for the tree and bark models used in figures 10 and 14, Oliver Deussen for the oak tree and bush models used in figures 8, 15, and 16, Keiichi Ohsaki for the tree models in figures 1 and 2, Brett Keating for the implementation of the algorithm in section 5 and the shadows in figures 17 and 18, Curtis Mobley for the Hydrolight code used in section 6, and Curtis Mobley and Brett Keating for reviewing this paper.

References:

[1] Przemyslaw Prusinkiewicz, Aristid Lindenmayer, James Hanan, "Developmental Models of Herbaceous Plants for Computer Imagery Purposes", Computer Graphics (Proceedings of SIGGRAPH 88). 22 (4), pp. 141-150, 1988.

[2] Przemyslaw Prusinkiewicz and Aristid Lindenmayer, The Algorithmic Beauty of Plants, Springer Verlag, New York, 1990.

[3] Radomír Mech, Przemyslaw Prusinkiewicz, "Visual Models of Plants Interacting With Their Environment", Proceedings of SIGGRAPH '96. pp. 397 - 410, 1996.

4] Oliver Deussen, Patrick M. Hanrahan, Bernd Lintermann, Radomír Mech, Matt Pharr, Przemyslaw Prusinkiewicz, "Realistic Modeling and Rendering of Plant Ecosystems", Proceedings of SIGGRAPH 98. pp. 275-286, 1998.

[5] Przemyslaw Prusinkiewicz, Lars Mündermann, Radoslaw Karwowski, Brendan Lane, "The Use of Positional Information in the Modeling of Plants", Proceedings of ACM SIGGRAPH 2001. pp. 289-300, 2001.

[6] Phillippe de Reffye, Claude Edelin, Jean Francon, Marc Jaeger, and Claude Puech, "Plant Models Faithful to Botanical Structure and Development", Proceedings of SIGGRAPH 1998, pp. 151-158.

[7] Jason Weber and Joseph Penn, "Creation and Rendering of Realistic Trees", Proceedings of SIGGRAPH 1995. pp. 119-128.

[8] William T. Reeves and Ricki Blau, "Approximate and Probabilistic Algorithms for Shading and Rendering Structured Particle Systems", Computer Graphics (Proceedings of SIGGRAPH 85). 19 (3), pp. 313-322, 1985.

[9] Jules Bloomenthal, "Modeling the Mighty Maple", Computer Graphics (Proceedings of SIGGRAPH 85). 19 (3), pp. 305-311, 1985.

[10] Bernd Lintermann and Oliver Deussen, "Interactive Modeling of Plants", IEEE Computer Graphics & Applications. 19 (1), pp. 56-65, 1999.

[11] K. Ohsaki, Y. Yamamoto, T. Suzuki, and H. Sayo, "A Representation Method for Needle-leaf-trees Based on Growth Models" (Japanese with English abstract), Graphics and CAD 52-4, pp. 19 - 26 (August 16, 1991).

[12] K. Ohsaki and T. Suzuki, "A Growth Model of Botanical Trees Having Abilities to Interact with the Light Environment" Japanese with English abstract), Graphics and CAD 65-6, pp 37-44 (October 22, 1993).

[13] Franklin C. Crow, "Shadow Algorithms for Computer Graphics", Computer Graphics 11 (2), pp. 242-248, 1977 (Proceedings of Siggraph 1997).

[14] Lance Williams, "Casting Curved Shadows on Curved Surfaces", Computer Graphics (Proceedings of SIGGRAPH 78). 12 (3), pp. 270-274, 1978.

[15] Turner Whitted, "An Improved Illumination Model for Shaded Display", Communications of the ACM, Vol. 23 (1980) pp. 343 - 349.

[16] Shechang Eric Chen and Lance Williams "View Interpolation for Image Synthesis", Proceedings of Siggraph 1993, pp. 279 - 288.

[17] Nelson Max and Keiichi Ohsaki, "Rendering Trees from Precomputed Z-Buffer Views", in "Rendering Techniques '95", (P. Hanrahan and W. Purgathofer, editors) Springer, Wien, 1995, pp. 74 - 81.

[18] Loren Carpenter, "The A-buffer, an Antialiased Hidden Surface Method", Computer Graphics (Proceedings of SIGGRAPH 84). 18 (3), pp. 103-108, 1984.

[19] Jonathan Shade, Steven Gortler, Li-wei He, and Richard Szeliski, "Layered Depth Images", Proceedings of Siggraph 1998, pp. 231 - 242.

[20] Nelson Max, "Hierarchical Rendering of Trees from Precomputed Multi-Layer Z-Buffers", in Rendering Techniques '96, (X. Pueyo and P. Schröder, editors) Springer, Wien, 1995, pp. 165 - 174.

[21] Nelson L. Max, Oliver Deussen, and Brett Keating, "Hierarchical Image-based Rendering using Texture Mapping Hardware", in Rendering Techniques '99 (D. Lischinski and G. W. Larson, editors) Springer, Wien, 1999, pp. 57 - 62.

[22] Gernot Schauffler, "Per-Object Image Warping using Layered Imposters", in Rendering Techniques '98 (G. Drettakis and N. Max, editors) Springer, Wien, 1998, pp. 145 - 146.

[23] Rüdiger Westermann and Thomas Ertl, "Efficiently Using Graphics Hardware in Volume Rendering Applications, Proceedings of Siggraph 1998, pp. 169 - 177.

[24] Nelson Max, "Atmospheric Illumination and Shadows", Computer Graphics 20(4) 1986 (Siggraph '86 Proceedings) pp. 117 - 124.

[25] Gary Watkins, "A real-time visible surface algorithm", PhD. Thesis, University of Utah (1970) UTECH-CSc-70-101.

[26] Nelson Max, "Anti-aliasing Scan Line Data", IEEE Computer Graphics and Applications, 10(1) 1990, pp. 18 - 30.

[28] Nelson Max, "Horizon Mapping: Shadows for Bump Mapped Surfaces", The Visual Computer 4(2) 1988 pp. 109 - 117.

[29] Nelson Max, "Light Beams" (video), Siggraph Video Review Issue 25 (1986) segment 9.

[29] Tomoyuki Hishita and Eihachiro Nakamae, "Half Tone Representation of Three Dimensional Objects Illuminated by Area Sources or Polyhedron Sources", IEEE Proceedings COMSAC (1983) pp. 237 - 241.

[30] Tomoyuki Hishita and Eihachiro Nakamae, "Continuous Tone Representation of Three Dimensional Objects Illuminated by Skylight", Computer Graphics 20(4) 1986 (Siggraph '86 Proceedings) pp. 125 - 132.

[31] Nelson Max, "Unified Sun and Sky Illumination for Shadows under Trees", CVGIP: Graphical Models and Image Processing 53(3) 1991 pp. 223 - 230.

[32] Michael Cohen and Donald Greenberg, "The Hemi-Cube: A Radiosity Solution for Complex Environments", Computer Graphics 19(3) 1985 (Siggraph '85 Proceedings) pp. 31 - 40.

[33] Nelson Max, "Sun and Shade" (video) Siggraph Video Review Issue 36 (1987) segment 8.Automatic Calculation of Soft Shadow Textures for Fast, High-Quality Radiosity

[34] Cyril Soler and François Sillion, "Automatic Calculation of Soft Shadow Textures for Fast, High-Quality Radiosity", in Rendering Techniques '98 (G. Drettakis and N. Max, editors) Springer, Wien, 1998, pp. 199-210.

[35] Cyril Soler and François Sillion, "Fast Calculation of Soft Shadow Textures Using Convolution", Proceedings of SIGGRAPH '98. pp. 321-332, 1998.

[36] Robert Cook, Thomas Porter, and Loren Carpenter, "Distributed Ray Tracing", Computer Graphics 18(3) 1984 (Siggraph '84 Proceedings) pp. 137 - 145.

[37] Dani Lischinski and Ari Rappoport, "Image-Based Rendering for Non-Diffuse Synthetic Scenes", in Rendering Techniques '98 (G. Drettakis and N. Max, editors) Springer, Wien, 1998, pp. 301 - 314.

[38] Brett Keating and Nelson Max, "Shadow Penumbras for Complex Objects by Depth-Dependent Filtering of Multi-Layer Depth Images" in Rendering Techniques '99, (D. Lischinski and G. W. Larson, editors), Springer, Wien, 1999, pp. 197 - 212.

[39] William Reeves, David Salesin, and Robert Cook, "Rendering Anti-aliased Shadows with Depth Maps", Computer Graphics 21(4) 1987 (Siggraph '87 Proceedings) pp. 283 - 290.

[40] Michael Cohen and John Wallace, "Radiosity and Realistic Image Synthesis", Academic Press Professional, 1993.

[41] Nelson Max, Curtis Mobley, Brett Keating, and En-Hua Wu, "Plane Parallel Radiance Transport for Global Illumination in Vegetation", in Rendering Techniques '99, (D. Lischinski and G. W. Larson, editors), Springer, Wien, 1999, pp. 239 - 250.

[42] Curtis Mobley, "Light and Water: Radiative Transfer in Natural Waters", Academic Press, San Diego, 1984.